The Pareto Archived Evolution Strategy : A New Baseline Algorithm for Pareto Multiobjective Optimisation

Joshua Knowles
Dept. of Computer Science
University of Reading
Reading RG6 6AY, UK.
J.D.Knowles@reading.ac.uk

David Corne
Dept. of Computer Science
University of Reading
Reading RG6 6AY, UK.
D.W.Corne@reading.ac.uk

Abstract-

Most popular evolutionary algorithms for multiobjective optimisation maintain a population of solutions from which individuals are selected for reproduction. In this paper, we introduce a simpler evolution scheme for multiobjective problems, called the Pareto Archived Evolution Strategy (PAES). We argue that PAES may represent the simplest possible non-trivial algorithm capable of generating diverse solutions in the Pareto optimal set. The algorithm is identified as being a (1 + 1) evolution strategy, using local search from a population of one but using a reference archive of previously found solutions in order to identify the approximate dominance ranking of the current and candidate solution vectors. PAES is intended as a good baseline approach, against which more involved methods may be compared, and may also serve well in some real-world applications when local search seems superior to or competitive with population-based methods. The performance of the new algorithm is compared with that of a MOEA based on the Niched Pareto GA on a real world application from the telecommunications field. In addition, we include results from experiments carried out on a suite of four test functions, to demonstrate the algorithm's general capability.

1 Introduction

Multiobjective optimisation using genetic algorithms has been investigated by many authors in recent years [2, 4, 5, 8, 9, 11, 12, 13]. However, in some real-world optimisation problems the performance of the genetic algorithm is overshadowed by local search methods such as simulated annealing and tabu search, either when a single objective is sought or when multiple objectives have been combined by the use of a weighted sum, e.g. see [10]. Unfortunately, little research has examined how local search methods can be applied to truly multiobjective problems (though see [6, 7]) in order to find diverse solutions in the Pareto optimal set. In this paper we introduce a novel evolutionary algorithm which employs local search for the generation of new candidate solutions but utilises population information in its selection procedure. We identify this algorithm as a $(1 + 1)$ evolution strategy [1], and suggest several extensions to it including how it may simply be converted to a $(\mu + \lambda)$ ES. Our algorithm, the Pareto Archived Evolution Strategy (PAES), extends the range of options open to the engineer interested in multiobjective optimisation and provides a baseline against which more involved algorithms can be compared. In addition, in some problems where delta-evaluation of computationally expensive objective functions can yield significant reduction in algorithm runtimes, it may be advantageous to use a local search method such as PAES.

The development of PAES arose from the investigation of a real world optimisation task in the field of telecommunications, involving the routing of calls through a sparse network. This task, which we describe in greater detail in Section 2, when posed as a *single* objective problem, can be solved very effectively using local search methods. In fact, we and other researchers have found simulated annealing and tabu search to be superior to or competitive with specialised EAs for searching the single-objective solution space [10]. However, a more natural representation of the task is as a three-objective problem with multiple Pareto optimal solutions (for which MOGAs are well suited). Nonetheless, given the efficacy of local search methods in the single-objective case, we ask whether local search can be extended to find diverse non-dominated solutions to this problem as well. To answer this question we developed PAES and compared its performance to a specialised MOEA which is based on the Niched Pareto Genetic Algorithm of Horn and Nafpliotis [8, 9]. Our results indicate that, on this problem, the PAES is able to generate a diverse set of good solutions comparable to the MOEA, and it does so in significantly less time.

The PAES algorithm is also compared to a steady-state version of the Niched Pareto Genetic Algorithm on a suite of four test problems. Three of these problems have been used by several researchers previously [2, 4, 8, 9, 12], and the fourth is a new problem devised by us as a further hard challenge to find diverse Pareto optima. The aim of this comparison is to generally explore and demonstrate the applicability of the PAES approach to standard multiobjective problems. A more thorough comparative analysis is the topic of future work, where we intend to make use of the multiobjective assessment techniques put forward by Fonseca and Fleming [3].

The main focus of this paper is to introduce and describe the PAES algorithm, and to identify its status within the multiobjective evolutionary algorithm continuum, indicating how it may be extended, and also making comparisons with similar, recently proposed multiobjective EAs. Particularly, we wish to argue that PAES represents the simplest possible, non-trivial Pareto multiobjective optimiser, and should thus serve the purpose of a good baseline algorithm against which others may be compared.

The remainder of this paper is organised as follows: Section 2 introduces the telecomms. routing problem, describes the objective function and provides background information regarding the performance of several algorithms on this task when it is posed as a single objective problem. In Section 3, we introduce PAES, describe its operation, and discuss how it may be used and extended in the future. Results are provided in Section 4. These include the solution sets found for the telecomms. routing task as well as results demonstrating the capability of PAES on a suite of four test problems. Finally, we summarise our findings in Section 5.

2 Off-line Routing in Circuit Switched Networks

The aim of this section is to describe the context in which the PAES algorithm was developed, thereby supporting the reader's interpretation of the results in Section 4. To this end, we briefly specify the problem, giving the objective function(s) used to compute solution quality. This is followed by a description of the heuristics we employ in the encoding and mutation operators. Finally, we report our findings on the performance of several algorithms on this task when it is cast in the form of a single-objective optimisation task (using a linear blend of the three underlying objectives).

Problem specification

The off-line routing task involves finding a set of routes for a given network/traffic combination such that communication costs and congestion are minimised. Its applications include the routing of traffic in synchronous digital hierarchy (SDH) networks (where clients book bandwidth in advance), re-routing of traffic on 'live' telecommunications networks where current traffic profiles are known, and also in the organisation of communications within a cluster for the computation of deterministic functions.

Our formulation of the problem is based on work by Mann and Smith [10] but differs in that we consider the problem over a series of discrete time steps in which traffic conditions change, whereas they considered the problem only in the static case. Our formulation of the problem is given below.

We are given a bandwidth limited telecommunications network over which we must route multiple traffic requests in such a manner as to achieve a feasible routing assignment, i.e. no link is over-capacitated (hard constraint). This is the primary objective. In addition, we impose a secondary objective that link utilisations should all be below a specified, fixed target utilisation. Finally, the routing assignment attempts to minimise the communications costs, costs being associated with the usage of each link.

Specifically, we are given a network $G = (N, E)$, where N is the set of n nodes and E is the set of m bidirectional edges. Associated with each edge $e \in E$ is a bandwidth capacity, $b(e)$, and a cost, $c(e)$. The bandwidth capacities of edges in the network $\{b(e)|e \in E\}$, lie in $\{16, 64\}$, that is there are two link types, a 'backbone' type of capacity 64 units and a 'local' type of capacity 16 units. The network exists in a time frame denoted by a set, T of l discrete time intervals, $t \in T$.

We are then given a set R of j communications which must be routed over G. Each communication, $r \in R$ specifies a source node $v(r)$ and a destination node $w(r)$, such that $\forall r\{v(r), w(r) \in N\}$. Associated with each communication, r, there is also a connection time, $\tau_\alpha(r) \in T$, a disconnection time, $\tau_\beta(r) \in T$, and a communication bandwidth $h(r)$.

The problem is to determine a path $P(r)$ in G for each communication, r, which connects $v(r)$ and $w(r)$ over the time interval $\tau_\alpha(r) \leq t < \tau_\beta(r)$, such that the objectives stated below are minimised. Note that there is only one path for each call, which exists over its entire time frame i.e. calls are not re-routed once they are connected.

We may now note that the total traffic on an edge e for the time interval t is given by

$$f(e,t) = \sum_{r \in R} \{h(r) \mid e \in P(r), \tau_\alpha(r) \leq t < \tau_\beta(r)\} \quad (1)$$

A viable routing is one in which the total traffic on each edge at each distinct time interval does not exceed the bandwidth capacity of that edge for that time interval, i.e.

$$\forall\, e \in E,\ \forall\, t \in T\ \{f(e,t) \leq b(e)\} \quad (2)$$

This can be achieved by minimising the deviation $f(e,t) - b(e)$, so long as $f(e,t) > b(e)$, i.e.

$$min. \sum_{t \in T} \sum_{e \in E} max\{f(e,t) - b(e), 0\} \quad (3)$$

which is our first objective.

A second objective is to find a minimum cost allocation of traffic through the network, satisfying constraint (3). The cost of routing one communication, $r \in R$, for one time interval, $t \in T$, on the path, $P(v, w)$, between v and w is given by

$$g(r,t) = \{h(r) \mid \tau_\alpha(r) \leq t < \tau_\beta(r)\} \times \sum_{e \in P(r)} c(e) \quad (4)$$

Hence, the *total* cost of routing all communications over the entire time frame, T, is

$$\sum_{t \in T} \sum_{r \in R} g(r,t) \quad (5)$$

Substituting (4) into (5) we have our second objective, which we wish to minimise

$$min. \sum_{t \in T} \sum_{r \in R} \left\{ \{h(r) \mid \tau_\alpha(r) \leq t < \tau_\beta(r)\} \times \sum_{e \in P(r)} c(e) \right\} \quad (6)$$

The third and final objective specified is to minimise the deviation from a target utilisation, u , for each link in the network, summed over all time steps i.e.

$$min. \sum_{t \in T} \sum_{e \in E} max \left\{ f(e,t) - \frac{u \times b(e)}{100}, 0 \right\} \qquad (7)$$

Thus, so long as $f(e,t) > (u \times b(e)/100)$, for at least one link $e \in E$, for at least one time step $t \in T$, there will exist some pressure on the optimisation process to find a more balanced solution. For our experiments we set $u = 50$. The three objectives defined above were initially combined into a single objective function using a weighted sum, and using weighting coefficients reported to give well balanced, feasible routing strategies [10].

Summary of Heuristics and Key Results

Following [10], we pre-calculate the K-shortest paths between each pair of nodes in the network, using a fast algorithm due to Yen [14]. A candidate solution vector (genotype) is then formulated so as to contain j K-ary elements i.e for each of the j communications, one of the K shortest paths associated with the source-destination pair of the communication is assigned. Solution vectors are initialised and mutated from a negative exponential probability distribution which biases them to strongly favour low values (and hence shorter paths).

To ensure fast solution evaluation we employ delta-evaluation of candidate solutions. This lends a great speed advantage to local search methods, which only change one or two elements in the solution vector at a time. Standard evolutionary algorithms typically cannot make good use of delta-evaluation, however, due to the large changes induced in recombination. To overcome this problem we devised a specialised crossover operator which only crossed one gene from one parent into the chromosome of the other, the particular gene chosen being a function of its allele value. We tested this specialised EA, together with a mutation only EA against a range of local search methods including standard hill-climbing, tabu search, simulated annealing and an evolution strategy.

Over a range of network sizes and traffic conditions we have found that the local search methods can converge to solutions of a given quality in significantly fewer evaluations than population based methods. Overall, we report that tabu search is the most effective algorithm, but there is little to choose between any of the local search methods we have tried. Our mutation-only EA performs worst, and an EA employing our specialised crossover operator is better but still requires significantly more evaluations than the local search algorithms to find equivalently satisfactory solutions. These findings led us to develop PAES, described next, enabling local search methods to be used for multiobjective optimisation.

3 The PAES Algorithm

Overview

The PAES algorithm was developed with two main objectives in mind. The first of these was that the algorithm should be strictly confined to local search i.e. it should use a small change (mutation) operator only, and move from a current solution to a nearby neighbour. This makes it quite different from most popular MOGAs, notably the Niched Pareto GA of Horn *et al.* [8, 9], Deb *et al.*'s Nondominated Sorting GA [13] and Schaffer's VEGA [12], which all maintain a population of solutions from which selection and breeding are carried out. The second objective was that the algorithm should be a true Pareto optimiser, treating all nondominated solutions as having equal value. Achieving both of these objectives together is problematic, however. This is because, in the majority of cases, when comparing a pair of solutions neither one will dominate the other. This problem is overcome in PAES by maintaining an archive of previously found nondominated solutions. This archive is then used as a means of estimating the true dominance ranking of a pair of solutions.

In fact, it is instructive to view PAES as comprising three parts: the candidate solution generator, the candidate solution acceptance function, and the Nondominated-Solutions (NDS) archive. Viewed in this way, PAES represents the simplest non-trivial approach to a multiobjective local search procedure. The candidate solution generator is akin to simple random mutation hillclimbing; it maintains a single current solution, and at each iteration produces a single new candidate via random mutation. Since the objective of multiobjective search is to find a spread of nondominated solutions, PAES necessarily needs to provide an NDS-list to explicitly maintain a limited number of these as and when they are found by the hillclimber. The design of the acceptance function is obvious in the case of the mutant dominating the current solution or vice versa, but troublesome in the nondominated case. Our approach is to learn from Horn *et al.*'s seminal work [8, 9], and hence use a comparison set to help decide between the mutant and the current solution in the latter case. The NDS-archive provides a natural and convenient source from which we can obtain comparison sets.

Arguably, even simpler multiobjective local search procedures are possible. One might have a simpler acceptance function, which always accepts the mutant unless the current solution dominates it. Or, it could only accept the mutant if it dominates the current. We have tried both of these, however, and found the results to be very poor. Echoing Horn *et al.*'s findings [8, 9], we find that the use of a non-trivially sized comparison set is crucial to reasonable results.

We must note that the idea of maintaining a list of nondominated solutions is not new. Parks *et al.* [11] recently describe a MOGA which also maintains an 'archive' of nondominated solutions. In their case, the overall algorithm is much more complicated than PAES. The archive is used as a repository for nondominated solutions, as in PAES, but also plays a key role as a pool of possible parents for selection. However, it is not used as a source of comparison to aid in the ranking of solutions. In terms of our 'three-part' view, their solution generator is a MOGA based on nondominated sorting, their archive is used as an extra source from which to fill the intermediate population, and their acceptance function simply replaces the current population with the newly constructed intermediate population in the normal way.

They found the use of this archive gave improved results over a traditional MOGA, tested on a particular application. They do not provide results (but indicate this as a future direction) on the use of their 'selective breeding' method on other or standard multiobjective test problems. Also beyond the scope of their published work is the use of the archive as an aid to estimating the fitness of population members (as in PAES). However, using the archive as a pool for selection is an interesting idea which we intend to incorporate in future population-based versions of PAES.

PAES

The structure of PAES is shown in Figure 1. The algorithm begins with the initialisation of a single chromosome (the current solution) which is then evaluated using the multiobjective cost function. A copy is made and a mutation operator is applied to the copy. This mutated copy is evaluated and forms the new candidate solution. The current and candidate solutions must then be compared. Acceptance is simple if one solution dominates the other but in the case where neither solution dominates, the new candidate solution is compared with a reference population of previously archived nondominated solutions. If comparison to the population in the archive fails to favour one solution over the other, the tie is split to favour the solution which resides in the least crowded region of the space.

The archive serves two separate purposes. First, it stores and updates all of the nondominated solutions (subject to diversity criteria) generated, ready for presentation at the end of a run. Second, during the run, it is used as an aid to the accurate selection between the current and candidate solution vectors by acting as an approximation to the current nondominated front. The latter is what provides the selection pressure, always pushing the process to find better solutions. Without this process, the algorithm is unable to differentiate between good and bad solutions with the result that it wanders rather aimlessly about the search space.

The archiving process is similar to that proposed in [11], but is not used as a pool for selection. The archive has a maximum size, $refpop$, which is set by the user to reflect the required number of final solutions desired. Each candidate solution generated which is not dominated by its parent (the current solution) is compared with each member of the comparison set. Candidates which dominate the comparison set are always accepted and archived. Candidates which are dominated by the comparison set are always rejected, while those which are nondominated are accepted and/or archived based on the degree of crowding in their grid location. This archiving and acceptance logic is made explicit in Figure 2.

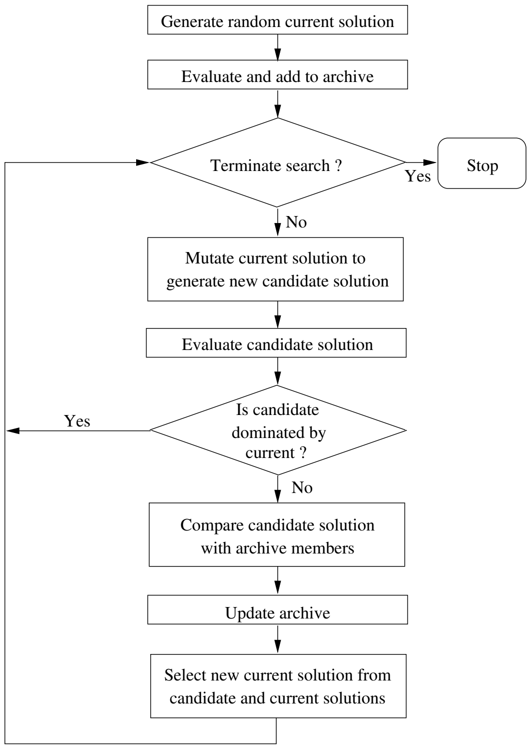

Figure 1: The Pareto Archived Evolution Strategy

To keep track of the degree of crowding in different regions of the solution space, a d-dimensional grid is used, where d is the number objectives in the problem. When each solution is generated its grid location is found using recursive subdivision and noted using a tree encoding. A map of the grid is also maintained, indicating for each grid location how many and which solutions in the archive currently reside there. When a candidate solution is in a position to join a full archive, it replaces one of the archived solutions with the highest grid-location count, so long as its own grid-location count is lower. This system is also used to select between the current and candidate solutions when the candidate is not dominated nor dominates any member in the archive. In this case the solution with the lower grid count is selected.

Future development

PAES may be identified as a $(1 + 1)$ ES and thus serves as a good baseline algorithm for multiobjective optimisation. However, we are also investigating the performance of $(1 + \lambda)$ and $(\mu + \lambda)$ variants of it. The former is identical to PAES$(1 + 1)$ except that λ mutants are generated from the current solution. Each is compared with the current solution via the archive as in PAES$(1 + 1)$, and the best of these replaces the current solution. The $(\mu + \lambda)$ version maintains a population of size μ from which λ copies are made, using tournament selection to select the fittest ones with respect to the archive. These copies are then mutated. Fitness is once

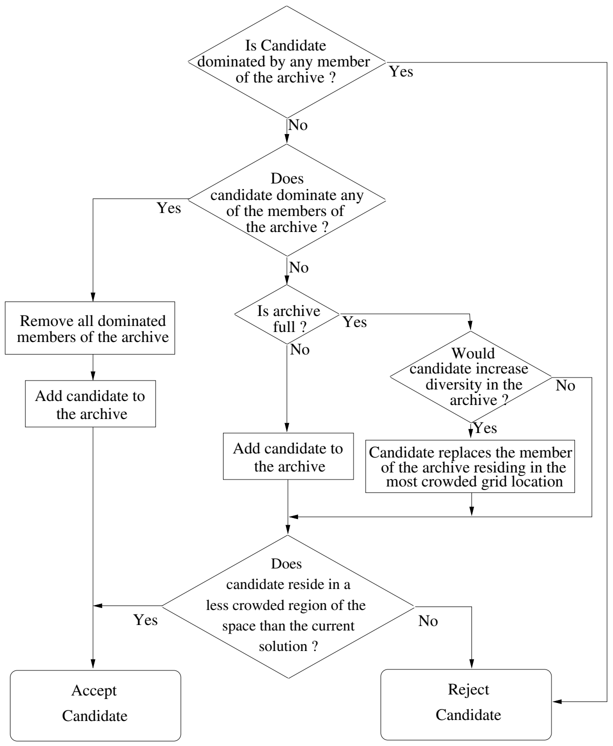

Figure 2: Archiving and Acceptance Logic

again assigned with respect to the archive. The fittest μ from the $\mu + \lambda$ solutions then replace the current population.

4 Results and Discussion

Off-line Routing Problem

We compared PAES with a non-generational EA based on the NPGA of Horn *et al.* [8, 9]. Here we present typical results from a single run on a problem in which 150 communications must be routed over a sparse network of 30 nodes.

PAES has only two parameters which must be set. These are the archive size, $refpop$ and the number of subdivisions of the space in the grid used for encouraging diversity. We have found both of these parameters easy to set on the problems tried. The archive size, $refpop$ is set to 100. At this level, the archive rarely fills up during a run of 20,000 evaluations. With $refpop$ set below 50, the performance of the algorithm begins to degrade, however. We recursively divide the phenotype space five times so that it is divided into $(2^5)^3 = 32768$ cubes. Each solution can be tagged with its grid location using just 15 comparisons with this setting.

Our NPEA uses tournament selection and a comparison set as described in [8, 9]. We investigated a number of different settings for population size, niche size, comparison set and tournament size. We found the best settings for the problem shown here to be $Pop = 150$, $t_{dom} = 20$, $Comparison\ set = 70$ and $\sigma_{share} = 0.01$ where the range of each objective is scaled to lie between 0 and 1.

Figure 3 shows a plot of the nondominated front found by the PAES and the NPEA on a run of 20000 evaluations. Where no solution has been found a default value has been assigned to the cost objective (z-axis). The surface found by PAES lies 'below' or at least on the same level as that found by the EA at all points, showing it has found better solutions. In addition, there is a large region in which the PAES has found solutions but where NPEA with a population of 150 has found none.

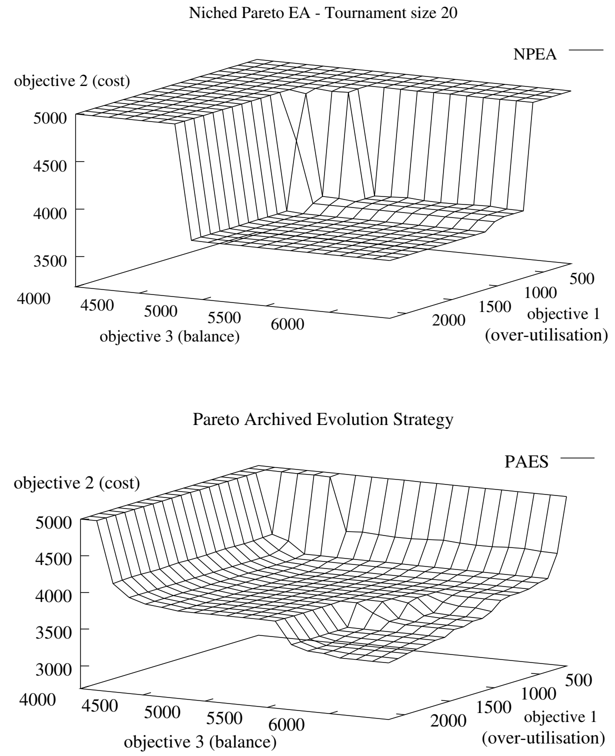

Figure 3: Nondominated surfaces obtained on an off-line routing task

Test problems

Following the potential shown by PAES on the off-line routing problem, we have compared PAES with the NPEA on a suite of standard test functions. The first four of these are the same as used by Bentley [2] i.e. Schaffer's functions F1, F2, and F3, and Fonseca's f1 [4], renamed here as F4. These functions (see figure 4) are now commonly used by researchers to test multiobjective optimisation algorithms. For reasons noted next we also designed a further test function which we call here F5. Functions F1–F4 contain, respectively, a single objective and one optima, a single range of optima, two ranges of optima, and a single range of optima spread across two dimensions. Their use in testing multiobjective optimisers is in their implicit setting of two challenges: First, the set of nondominated solutions delivered by the optimiser should contain all of the function's Pareto optima. Second, it is generally felt best if there is no strong bias favouring one Pareto optimum over others. In other words, in a MOEA, for example, the number of copies of each Pareto optimum in the final population should be similar. If not, this would seem to reveal a bias which may be undesirable in practical applications.

We designed F5 (described below) to provide stronger challenges in these respects; it is easily defined, but is a non-trivial problem. Each Pareto optimum is intrinsically difficult to find for an EA, and there are n distinct Pareto optima for chromosomes of length n, each of which has a different frequency i.e. some are far easier to find than others. This makes both challenges (as described above) stringent tests for any multiobjective optimiser.

The function F5 uses an n-ary chromosome of n genes. There are two objectives defined by the following two functions.

$$f_{51} = n - 1 - \sum_{i=0}^{n-2} \begin{cases} 1 & \text{if } G_{i+1} - G_i = 1 \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

$$f_{52} = n - 1 - \sum_{i=0}^{n-2} \begin{cases} 1 & \text{if } G_i - G_{i+1} = 1 \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

where G_i is the ith gene. We used $n = 16$ in our experiments.

In all experiments, we compared the NPEA with PAES. For each case, preliminary parameter setting investigations were conducted to find reasonably good values for the NPEA, and the results recorded are for those settings. PAES is the same each time, with a maximum archive size of 100. NPEA always had a population size of 100, which seemed suitable for comparison with the similar setting for PAES.

On problem F1, both PAES and the NPEA converge to the single optimal solution, as expected. Figures 5–8 indicate the performance of PAES and the NPEA on the remaining test functions. On each problem, both algorithms were run 20 times for the same number of function evaluations. For PAES, the final archive list was recorded, whereas for the NPEA the recorded solutions were the population in the last generation. Thus, the total frequency distribution of solutions as seen by a user for each algorithm is plotted. In general, the frequencies of points for the NPEA are higher than PAES but this is due to the fact that repeated phenotypes are always removed from the archive list in PAES so that usually fewer than 100 solutions are returned per run.

On problem F2, both algorithms find solutions across the whole P-O range. PAES returns fewer solutions outside this range, as one might expect, because it is returning only the nondominated solutions which it has stored in its archive. The distribution is somewhat noisier with PAES but seems to be less biased than the NPEA.

Similarly, on problem F3 PAES gives a clean, if rather noisy, distribution of points within the two separate Pareto optimal ranges and very few sub-optimal solutions. Its performance seems to compare well with NPEA, which gives relatively few results in the range 1.8–2.0 and 4.0–4.2.

On problem F4 both algorithms perform well, spreading solutions across the entire P-O range. However, the NPEA

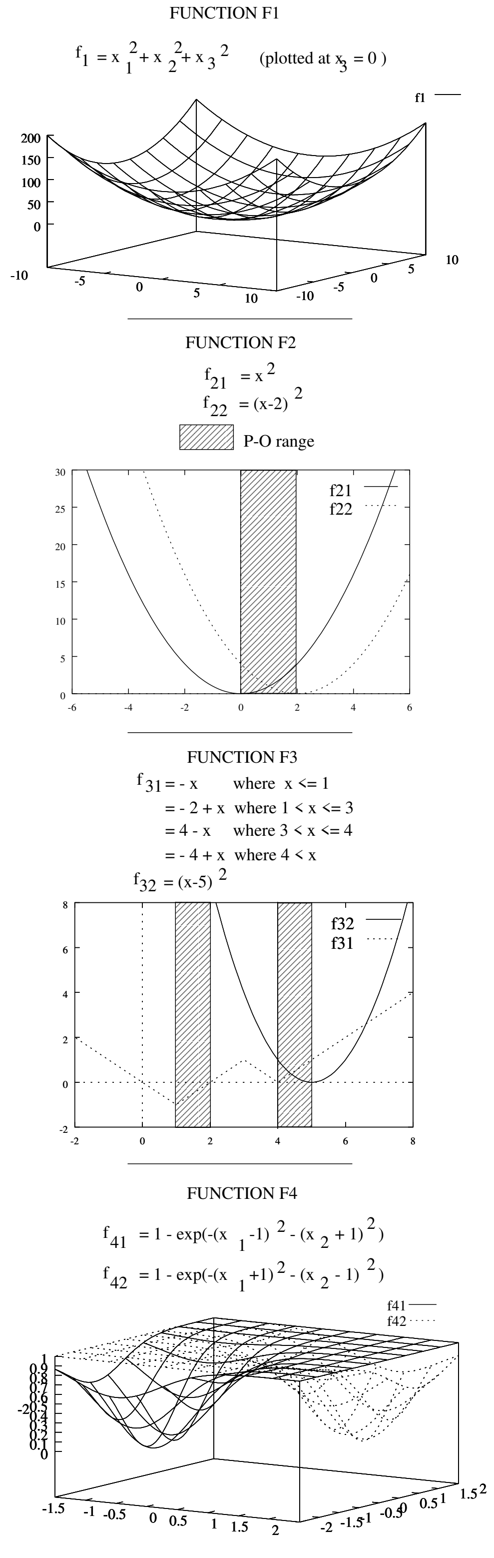

Figure 4: Test functions F1 to F4

distribution reveals that it finds very few solutions at the tradeoff point1 $x1 = 0.0, x2 = 0.0$. It is also far more biased towards solutions centred around [0.5, -0.5] and [-0.5, 0.5] . PAES, by contrast, generates an extremely clean and accurate

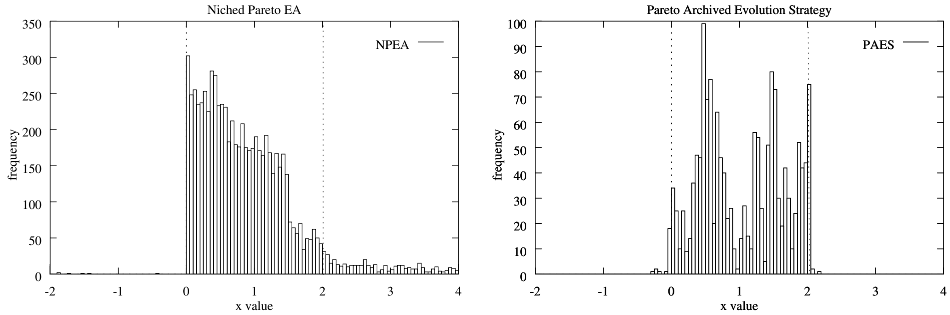

Figure 5: Test results - F2

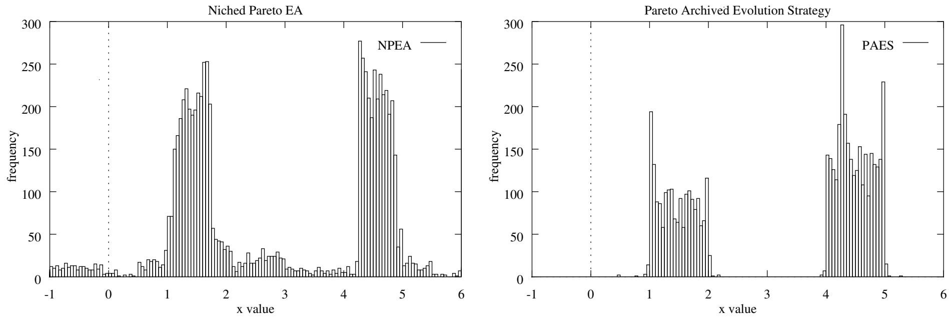

Figure 6: Test results - F3

distribution with little bias away from the central point [0.0, 0.0]. The distribution is also tighter, concentrating closer to the true P-O range which lies on the straight line extending from [1.0, -1.0] to [-1.0, 1.0].

Both algorithms had trouble with problem F5, finding few P-O solutions (having $f51 + f52 = 15$) at the edges of the range, in 15000 evaluations. Both generate a good spread of solutions with $f51 + f52 = 16$ or 17 but PAES is more successful on this problem than the NPEA. Its whole distribution is shifted slightly towards the optimum range and it has longer tails showing that it has found more of the difficult solutions. This is somewhat surprising on this difficult problem which one might suppose is more suited to a hyper-plane sampling algorithm (employing one-point crossover) than a local search algorithm such as PAES.

5 Conclusion and Future Work

We have described PAES, which in its (1+1)-ES form can be viewed as a simple baseline technique for multiobjective optimisation. When used to address the multiobjective form of the off-line routing problem, (1+1)-PAES provided results strongly competitive with a MOEA. When then compared on a suite of test functions (1+1)-PAES was again found to be strongly competitive with a MOEA in each case. In terms of speed, coverage of the Pareto tradeoff frontier, and general absence of strong bias within that frontier, (1+1)-PAES appears to perform consistently well, despite being essentially a simple algorithm.

However, certain disclaimers and qualifications are necessary. First, it could well be that further parameter-setting investigation of the NPEA, and also the MOEA used in the off-line routing experiments, might yield better results than we have found here. Further, several alternative MOEA algorithms could be tried, such as Deb *et al.*'s nondominated sorting algorithm [13] as well as the strongly elitist MOEA put forward by Parks *et al.* [11], also employing an archive of nondominated solutions.

In the absence of such further comparative studies, we can tentatively suggest that (1+1)-PAES is a powerful stand-alone strategy for multiobjective optimisation problems. However, if comparisons with alternative MOEAs turn out to leave (1+1)-PAES floundering in their wake, we believe it remains true that (1+1)-PAES is a good baseline algorithm against which such methods may be compared, and also that it represents a good foundation upon which to design more sophisticated alternative MOEAs. The latter is currently under investigation, whereby we are exploring $(1 + \lambda)$ and $(\mu + \lambda)$ PAES. Preliminary results suggest that this algorithm can outperform (1+1)-PAES on certain problems.

Acknowledgments

The first author would like to thank BT Labs Plc. for the continuing sponsorship of his PhD.

Bibliography

[1] Bäck, T. (1996) Thomas Bäck, Evolutionary Algorithms in Theory and Practice, Oxford University Press, 1996

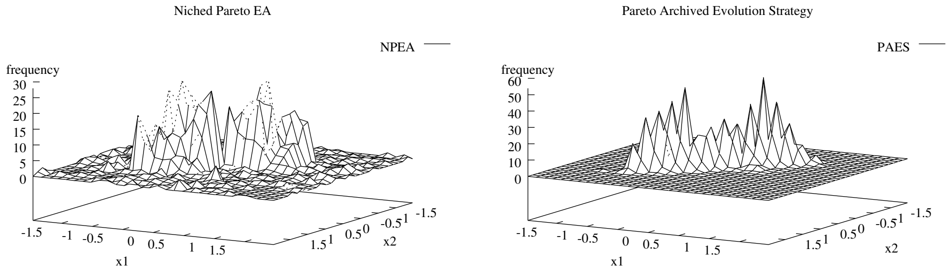

Figure 7: Test results - F4

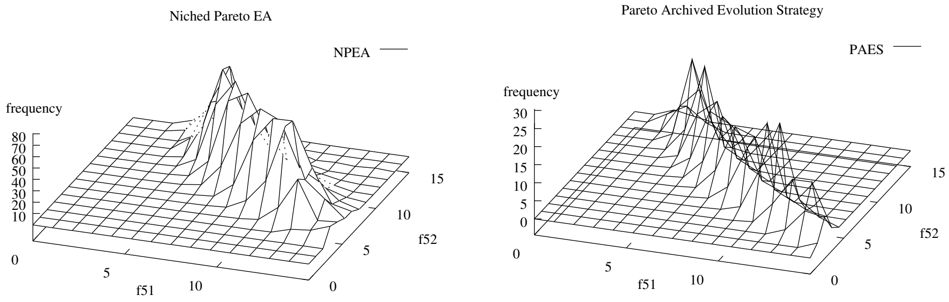

Figure 8: Test results - F5

[2] Bentley, P.J., Wakefield, J.P. (1997) Finding Acceptable Solutions in the Pareto-Optimal Range using Multiobjective Genetic Algorithms. Presented at the 2nd On-line World Conference on Soft Computing in Engineering Design and Manufacturing (SC2), 23-27 June 1997.)

[3] Fonseca, C., Fleming, P.J. (1995) An Overview of Evolutionary Algorithms in Multiobjective Optimisation. Evil. Comp. **3**, 1-16.

[4] Fonseca, C.M., Fleming, P.J. (1995) On the Performance Assessment and Comparison of Stochastic Multiobjective Optimisers. Parallel Problem Solving from Nature. In H-M. Voigt, W. Ebeling, I. Rechenberg, and H-P. Schwefel (eds.), Parallel Problem Solving From Nature - PPSN IV, Springer LNCS volume 1141, 1996, pp. 584-593.

[5] Goldberg, D.E. (1989) Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley.

[6] Hansen, M.P. (1996) Tabu Search for Multiobjective Optimization : MOTS. Presented at MCDM '97, Cape Town, South Africa, Jan 6-10.

[7] Hansen, M.P. (1997) Generating a Diversity of Good Solutions to a Practical Combinatorial Problem using Vectorized Simulated Annealing. submitted to Control and Cybernetics, August 1997.

[8] Horn, J., Nafpliotis, N., Goldberg, D.E. (1994) A niched pareto genetic algorithm for multiobjective optimization. Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, Volume 1. Piscataway, NJ: IEEE Service Centre, 67-72.

[9] Horn, J., Nafpliotis, N. (1994) Multiobjective Optimisation Using The Niched Pareto Genetic Algorithm. IlliGAL Report 93005, Illinois Genetic Algorithms Laboratory, University of Illinois, Urbana, Champaign.

[10] Mann, J.W., Smith, G.D. (1996) A Comparison of Heuristics for Telecommunications Traffic Routing. Published in Modern Heuristic Search Methods, Editor V.J.Rayward-Smith, I.H.Osman, C.R.Reeves and G.D.Smith. John Wiley and Sons Ltd.

[11] Parks, G.T., Miller, I. (1998) Selective Breeding in a Multiobjective Genetic Algorithm. Parallel Problem Solving from Nature : 5th international conference proceedings, Agoston E. Eiben (ed.) Springer.

[12] Schaffer, J. D. (1985) Multiple objective optimization with vector evaluated genetic algorithm. Proceedings of the 1st International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, Inc., San Mateo, 93-100.

[13] Srinivas, N., Deb, K. (1994) Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation, 2(3), 221-248.

[14] Yen, J.Y. (1971) Finding the K Shortest Loopless Paths in a Network. Management Science Vol.17 No.11, July 1971.